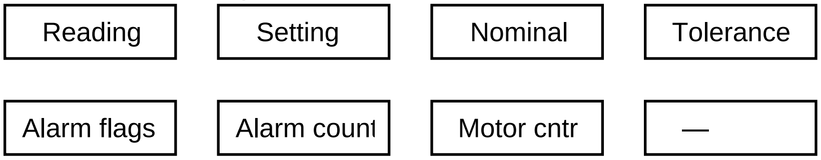
# System Tables

*An introduction to their uses*

Sep 27, 1990

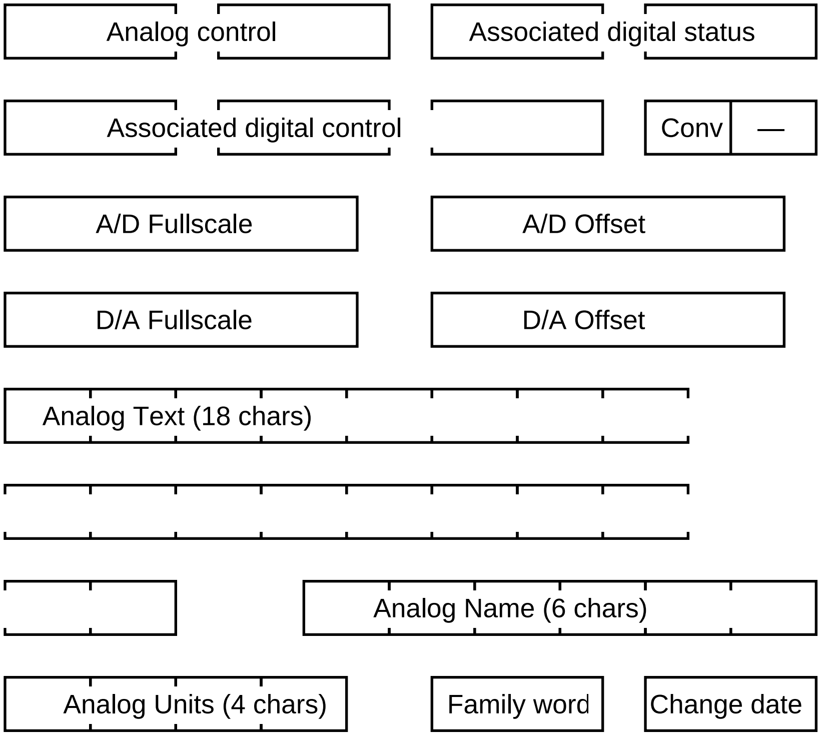
The VME Local Station software makes heavy use of system tables, which are statically allocated arrays of records mostly in non-volatile memory. Each table is specified in the system table directory, whose entries are indexed by system table#. This note introduces each system table and describes its use.

| # | NAME | Description | Indexed by | #bytes/entry |
|---|---|---|---|---|
| 0. | ADATA | Analog Data | CHAN | 16 |



Analog channel dynamic values are kept here. The latest analog reading, the last setting successfully issued and the analog alarm scan parameters comprise this entry. The motor countdown word is also used for capture data. (See Data Access Table entry type $16.) This table is the analog data pool. See the document "Alarms Task" for more details about Alarm flags.

| | | | | |
|---|---|---|---|---|
| 1. | ADESC | Analog Descriptor | CHAN | 64 |



The analog local static database includes these fields for each channel. Additional fields used for D0 alarms are in AADIB. The date-of-last-change for this entry is encoded into 16 bits as year(7)/month(4)/day(5).

See the document "Analog Control Types" for details on the analog control field. See the document "Digital Control Pulse Delays" for more details on Associated status and control bits. See the document "Related Groups of

Database” for more about the local database in general. (The latter document is fairly old, however.)

2. BALRM Binary Alarm BIT 4

| | |
|---|---|
| Alarm flags | Alarm count |

The alarm flags for each binary Bit are the same as those for an analog channel, except that bit#14 of the word is used for the nominal bit state.

3. BDESC Binary Bit Titles BIT 16

| |
|---|
| Binary Text (16 chars) |

This is a 16-char text description of each binary Bit.

4. RDATA Read Data Access Table Indx 16

| | | | |
|---|---|---|---|
| Type \| Tbl# | Entry# | Pointer | |
| — | — | — | Count |

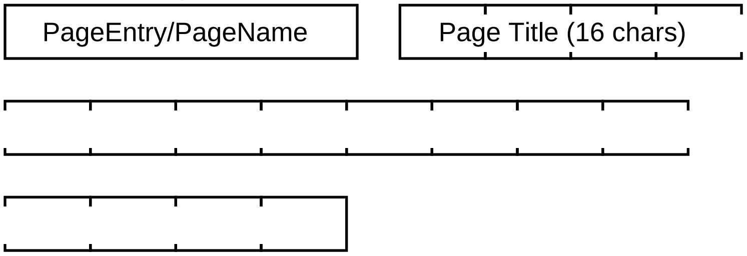
This entry is difficult to illustrate since there are so many formats used depending upon the Type byte. See “RDATA Formats” and “RDATA Periodicity” documents for many more details.

5. BBYTE Binary Status Bytes BYTE 1

| |
|---|
| status |

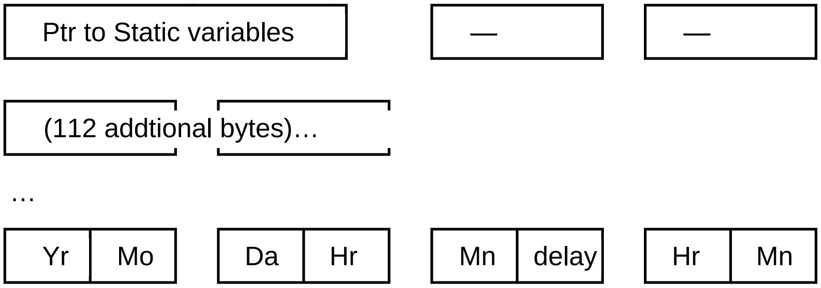
Each entry is a single byte that is read (via the $04 Data Access Table entry) based upon the contents of the corresponding entry in the BADDR table. This table constitutes the binary data pool.

6. PAGEP Page Pointer PAGE 20



The first 4 bytes may be either a pointer to the entry point for the page application or the 4-char name which, when prepended with PAGE, forms the named program key to the CODES table entry that refers the page application program in non-volatile download and volatile executable memory. The 16-character title displayed for this page is also included here. See the document “Local Station Applications” for a list of page applications and see separate user guide documents about each one.

7. PAGEM Page Memory PAGE 128



Each page application is given 120 bytes of page-private memory that is saved across display page invocations. The example shows the first 4 bytes being used to house a pointer to a static variables record that is used for variables whose values must be saved across successive cycles while the display page is active. The last 8 bytes of the 128-byte entry are used for the auto-page parameters, which allow automatic and periodic invocation of a display page. The first 5 bytes hold the Next time the page will be called up, while the last 2 bytes specify a Delta periodic time, which may be 0000 for a one-shot auto-invocation. The delay byte specifies a timeout (in the range one 15 Hz cycle to 4 minutes) after which the system will automatically terminate the automatic invocation of the page. See the document “Auto-page on Demand” for more details on the auto-page facility.

8. `LISTP` Active Data Request Ptr Indx 8

| | | | |
|---|---|---|---|
| Short | Main | Last | #active |
| — | — | — | — |

The header of this table includes an offset to the Short and Main sets of entries as well as the offset to the Last vacant entry used in the Main set. The number of active requests includes locally-generated and data server requests, but it does not include ordinary network data requests.

| | | | |
|---|---|---|---|
| List# | Usage Cntr | — | — |

The Short set portion of the table includes the “real” 11-bit List# associated with the byte size list# specified by the ReqData caller followed by a usage count of this associated byte size list#.

| | | |
|---|---|---|
| — | Usage Cntr | Ptr to request block |

The Main set portion of the table includes a usage counter of this table entry and a ptr to its active request support memory block. The latter is zero if the entry is inactive. This table is for internal system use only.

9. `CODES` Downloaded Programs Indx 32

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| T | Y | P | E | | N | A | M | E |

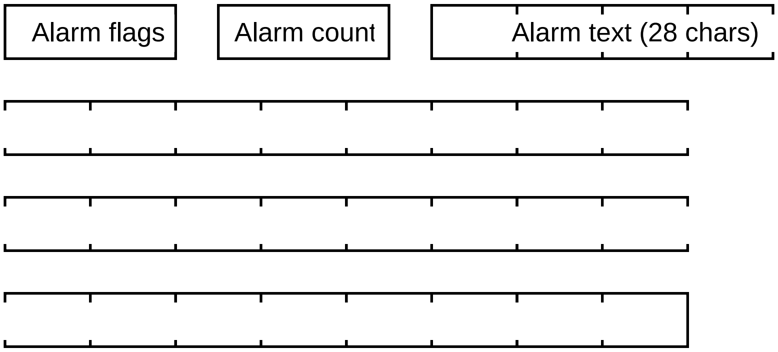
| | |
|---|---|
| Size | Checksum |
| Ptr to downloaded copy | Ptr to executable copy |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Yr | Mo | Da | Hr | Mn | Sc | | Copy cntr |

Each entry denotes a named downloaded program. The first 4 chars are used to indicate different types of programs; *e.g.*, `PAGE` is used for page applications and `LOOP` is used for local applications. Within a type, a 4-char name mnemonically denotes a particular program code. Each valid entry contains a ptr to the downloaded copy in non-volatile memory and an optional ptr to volatile executable memory used while the program is active. The download time is recorded along with a diagnostic count of the number of times that space is allocated in the executable area for a copy of the downloaded program.

10. CDATA Comment Alarm Data

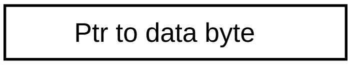
CMNT 32



This table includes the static text that is used for Comment alarms along with the usual alarm flags and count to permit bypassing such messages. As of this writing, the only Comment alarm is a system reset message; this table makes provisions for more such alarm messages.

11. BADDR Binary Byte Address

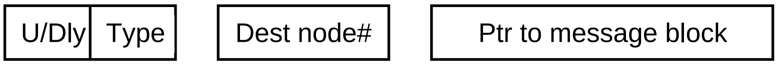
BYTE 4



The Data Access Table entry type $04 causes all entries of this table to be scanned and each data byte read placed into the BBYTE table. If the hi byte of the ptr has the value $80, the lo 24 bits are considered a ptr into a one-word 1553 command block data byte.

12. OUTPQ Output Pointer Queue

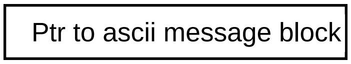
Indx 8



This table includes an entry for every message that is destined for the network. The sign bit of the first byte is a “used” bit that is set when the entry has been passed on to the network. The other 7 bits are a delay counter used for a timeout on network response.

13. PRNTQ Serial Output Queue

Indx 4



The message block includes one line of ascii text to which a CR-LF is appended. This queue allows spooling of lines of text for serial output.

14. LATBL Local Applications

Indx 32

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| — | eStat | Call cntr | | N | A | M | E |
| Ptr to static variables | | | | eBit# | | — | |
| — | | — | | — | | — | |
| — | | — | | — | | — | |

Local applications are the means of supporting closed loops in the local station. Each entry characterizes an instance of a closed loop, allowing for multiple uses of the same closed loop code with different parameters. The code to be invoked is known by 4-char name. When it is called, a ptr to the last 24 bytes of this entry is passed. This provides a means of storing a ptr for the static variables used by this instance for use in subsequent calls while the loop is active. The eBit# is the enable/disable Bit# used for control of the closed loop instance. The eStat is the memory of its last value to detect changes in the enable/disable state. The Call cntr is a diagnostic count of the calls made to the code.

15. CPROQ Co-processor Queue Ptrs

COP 16

| | | | |
|---|---|---|---|
| Ptr to command queue | | Size | — |
| Ptr to readback queue | | Size | — |

A co-processor is a cpu that shares the same backplane with the local station. A simple one-way message queue is used for communication with the CoP. Provision is made in the entry for a read-back queue, but as of this writing, no system support yet exists for such.

16. MMAPS Memory-mapped

Template Indx 8

| | | | | | |
|---|---|---|---|---|---|
| M | M | Directory | #entries | Last err# | |
| Last address ident used | | | — | — | |

This table supports gather-read and scatter-write access to memory according to commands in a template. Multiple memory layouts can be supported by separate entries. The header of the table includes a key to recognize a valid MMAPS table, an offset to the start of the directory portion of the table, The #entries in the directory and some error diagnostics.

| | | | |
|---|---|---|---|
| Cmds offset | #reads | #writes | #errors |

For each command, the type of command entry is followed by the offset to the data to access, the #bytes to access and the #bytes to skip. For the loop type command entry, the following layout applies:

| | | | |
|---|---|---|---|
| D 0 D C | #cmnds | loopCnt | — |

The #cmnds is the number of commands following to be looped over. See the document “MMAPS Table Entries for D0 Boards” for more details.

17. Q1553 1553 Controller Queue
Ptrs C1553 4

| |
|---|
| Ptr to 1553 cmd queue |

For each 1553 controller is use, a command queue is initialized and a ptr to it placed into this table. The queue itself is in the memory on the 1553 controller board, currently at offset $F000 from the start of a 64K block. (If the controller has only 16K memory, it is effectively at offset $3000 from the start.) See the document “1553 Data Acquisition” for more details about this command queue. Also see the document “VME 1553 Interface” for more about 1553 command blocks and related topics.

18. DSTRM Data Streams
STRM 32

| | | | | |
|---|---|---|---|---|
| qFlags | qType | eSize | hSize | qSize |
| qPtr | | | – | – |
| data stream 8-character name | | | | |
| – | | – | – | – |

There is an entry in this table for each data stream in the system, indexed by data stream number. At reset time, the data stream queues are initialized according to valid `DSTRM` entries. Only qType=1 is currently supported. The eSize word allows for fixed or variable size queue packets. The hSize word is the size of the *data stream-specific* header component of the queue header. The qSize includes the space for the queue header. The qPtr points to the data stream queue header. See “Data Streams Implementation” for more details.

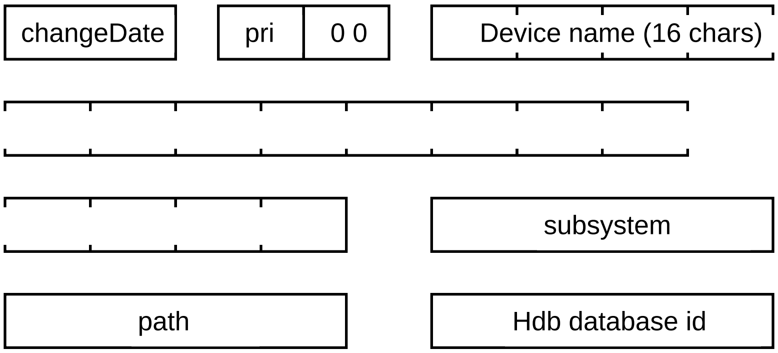
19. SERIQ Serial Input Queue
— 1K

| | | | | |
|---|---|---|---|---|
| OUT2 | OUT1 | | IN | LIMIT |
| START | CRCnt | chCnt | errCnt | lineCnt |

The header for this queue includes offsets for the first and second “output offset. Nulls and LFs are ignored. When a CR has been read and placed into the queue, an event is sent to activate the Serial Task. It advances the `OUT1` offset of the queue. A user requesting listype #36 data advances the `OUT2` offset, this consuming the data read. (A data stream implementation of serial port access could eliminate this side effect.) See the document “VME System Serial Port Handling” for more details.

20. ----- spare

21. `AADIB` Analog Alarm Device Info
Blocks `CHAN` 32



This table supports auxiliary information required by D0 alarm messages for each analog channel The last 30 bytes of each table entry is downloaded from the Host’s database. A date-of-last-change is automatically recorded in the first word in the 16-bit format of Year(7)Month(4)Day(5). See the document “D0 Alarms” for more details.

22. `BADIB` Binary Alarm Device Info
Blocks `BIT` 32

(See `AADIB` for layout.) This table supports auxiliary information required by D0 alarm messages for each binary bit.

23. `CADIB` Comment Alarm Device
Info Blocks `CMNT` 32

(See `AADIB` for layout.) This table supports auxiliary information required by D0 alarm messages for each comment alarm index.

24. ----- spare

25. ----- spare

26. ----- spare

27. ----- spare

Indx 16

| | | | |
|---|---|---|---|
| IN | LIMIT | — | — |
| — | — | — | — |

The header of this *optional* diagnostic queue includes the offset to the next entry to be placed and the total size of the queue.

| | | |
|---|---|---|
| tag | size | ptr to allocated block |
| cycleCnt | msec | caller's return address |

Each entry denotes the occurrence of a call to Alloc or to Liber to allocate or free a block of dynamic memory. The tag word has values of `$AA00` for an Alloc call and `$FF00` for a Liber call. The size of the block allocated or freed and the ptr to the block follow. The time of the call is specified as a count of 15 Hz cycles since reset and a count of 0.5 msec since the start of the current cycle. The last longword is a copy of the return address of the call to Alloc or Liber.

30. `TRING` Token Ring Network
— 8K

All token ring network functions are organized around the use of this table, the structure of which is fairly complicated. Its layout is presented in some detail in the document “VME Token Ring Table.”